2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC)
October 6-10, 2024. Sarawak, Malaysia

Surrogate-assisted Multi-class Collaborative Teaching and Learning Optimizer for High-dimensional Industrial Optimization Problems

Jing Bi[1,2], Ziqi Wang[1,2], Haitao Yuan[3], Jinhong Yang[4] and Jia Zhang[5]

[1]College of Computer, Beijing University of Technology, Beijing 100124, China
[2]Beijing Laboratory of Smart Environmental Protection, Beijing University of Technology, Beijing 100124, China
[3]School of Automation Science and Electrical Engineering, Beihang University, Beijing 100191, China
[4]CSSC Systems Engineering Research Institute, Beijing, China
[5]Dept. of Computer Science in Lyle School of Engineering, Southern Methodist University, Dallas, TX, USA

2024 IEEE International Conference on Systems, Man, and Cybernetics (SMC) | 978-1-6654-1020-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/SMC54092.2024.10831544

Abstract—**Swarm intelligence and evolutionary algorithms are widely applied in industrial scheduling, mobile edge computing, *etc* due to their strong robustness and fast optimization speed. However, some real-world industrial optimization problems involve numerous decision variables, known as high-dimensional problems. Current algorithms often require considerable computational resources to evaluate objective function values because of high-dimensional decision spaces. Moreover, they are also prone to be trapped in local optima. To solve the above problems, this work proposes an improved algorithm named Surrogate-assisted Multi-class Collaborative Teaching and learning optimizer (SMCT). A multi-class collaborative teaching and learning optimizer is proposed as a base optimizer to improve exploration and exploitation abilities. Furthermore, an autoencoder-assisted radial basis function is proposed as the surrogate model to replace true function evaluations, thereby saving computational resources and balancing the complexity and accuracy in fitting true models. Finally, experimental results demonstrate that SMCT surpasses its existing peers in both search accuracy and convergence speed across eight high-dimensional benchmark functions.**

Index Terms—**Meta-heuristic optimization algorithms, autoencoders, surrogate models, high-dimensional problems, and radial basis functions.**

I. INTRODUCTION

Swarm intelligence and evolutionary algorithms are widely applied in many fields, *e.g.*, system scheduling optimization [1]–[3], cybernetics [4], and mobile edge computing [5]–[7]. Traditional optimization methods, *e.g.*, Newton's methods [8] and simplex algorithms need to traverse entire decision spaces, which cannot entirely explore them in a short time and they suffer from combinatorial explosion [9]. Motivated by the social and natural phenomena of biological groups, many optimization algorithms are designed to solve the above complex optimization problems, *e.g.*, grey wolf optimization (GWO) [10], teaching-learning-based optimization (TLBO) [11] and genetic algorithm (GA) [12]. They are simple to implement and highly generalized, making them applicable to a wide range of optimization problems. However, many complex combinatorial industry problems have a great number of decision variables [13]. Given the complexity, non-linearity, constraints and many other characteristics of high-dimensional problems, traditional optimization algorithms cannot well cover their whole decision spaces. Moreover, function evaluations become increasingly expensive [14], and the optimization algorithms need a huge number of function evaluations, consuming a large number of computational resources during their optimization processes [15]. Thus, it is essential to propose an optimizer that can find the global optima efficiently with fewer computational resources.

As function evaluations in high-dimensional problems become expensive, the introduction of surrogate models alleviates this problem [16]. By learning the landscape features of a model, surrogate models can substitute a part of it in function evaluations [17], thereby saving computational resources. However, the training of surrogate models under high-dimensional data also brings additional computational costs. As a result, it becomes essential to balance the accuracy and complexity of the surrogate models. An excessively complex surrogate model consumes too many resources, but an inaccurate one may mislead the optimization direction, leading to inaccurate or even false optimization results [18]. Therefore, additional strategies need to be considered to ensure the validity of the final optimization results.

To solve the above problems, an improved Surrogate-assisted Multi-class Collaborative Teaching and learning optimizer (SMCT) is proposed. A Multi-class Collaborative Teaching and learning optimizer (MCT) is first designed as the base optimizer to improve exploration and exploitation abilities. Specifically, the population is split into multiple classes in the teaching phase, and each class has its corresponding teacher. Learners converge to their teacher in different classes. In that case, multiple classes can better cover the decision space and prevent the algorithm from developing premature convergence problems. In the learning

This work was supported by National Natural Science Foundation of China (NSFC) under Grants 62173013 and 62073005, Beijing Natural Science Foundation under Grant 4232049, and Fundamental Research Funds for Central Universities under Grant YWF-23-03-QB-015.

phase, the GWO's search strategy is incorporated and multiple classes are recombined into a single population again. The top three best learners are selected and the population approaches them with the hunting strategy of GWO. Moreover, MCT adopts GA's mutation strategy and simulated annealing (SA)'s Metropolis acceptance criteria. The former allows fluctuations of learners' performance and the latter allows acceptance of worsened individuals, both of which facilitate MCT to step out of local optima. Furthermore, an Autoencoder-assisted Radial Basis Function (ARBF) is designed as the surrogate model, which incorporates an autoencoder to enable the radial basis function (RBF) to be trained in a low-dimensional space. Therefore, RBF can be trained with fewer computational resources and shorter training time. Furthermore, SMCT employs a reevaluation strategy to ensure the accuracy of optimization results. To be specific, some of the individuals are reevaluated by the true model. Finally, we compare SMCT with its three typical peers by using eight high-dimensional benchmark functions, and the simulation results show the superiority of the SMCT.

II. PROPOSED FRAMEWORK

A. *Multi-class Collaborative Teaching and learning optimizer (MCT)*

In MCT, individuals in the population are learners. Moreover, the teacher has the best fitness value, and the remaining individuals are learners that approach the teacher in the teaching phase. The fitness value of each learner represents its learning performance. However, the teacher has a significant impact on the optimization direction of the whole population. Once the teacher falls into a local optima, learners are similarly susceptible to encountering it as they move closer to the teacher. As a result, in the teaching phase of MCT, learners are split into multiple classes, each of which owns a teacher and multiple learners moving closer to their teacher. N denotes the number of random individuals (X_1, X_2, $\cdots$, X_N) in a decision space. N individuals are given as:

$$\begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix} = \begin{bmatrix} X_1^1, & X_1^2, & \cdots, & X_1^D \\ X_2^1, & X_2^2, & \cdots, & X_2^D \\ \vdots & \vdots & \ddots & \vdots \\ X_N^1, & X_N^2, & \cdots, & X_N^D \end{bmatrix} \tag{1}$$

where D is the dimension of the population, and $X_i^d \in [\check{b}_d, \hat{b}_d,]$, $1 \le d \le D$ and $1 \le i \le N$. $\check{b}_d$ and $\hat{b}_d$ denote lower and upper bounds of dimension d. Moreover, X_i represents individual i and X_i^d denotes its dimension d. Individuals are updated based on the average position of the population (M_d) in the teaching phase, *i.e.*,

$$\begin{gathered} X_i^d(t+1) = X_i^d(t) + r_d \left(X_T^d(t) - T_f M_d(t) \right) \\ M_d = \frac{1}{N} \sum_{i=1}^{N} X_i^d \\ T_f = \mathbf{round}[1 + \mathbf{rand}(0,1)] \end{gathered} \tag{2}$$

where $X_i^d(t)$ denotes the value of dimension d of individual i in iteration t. Moreover, $X_T^d(t)$ represents the value of dimension d of the teacher in iteration t, and r_d is a random value in [0,1]. Moreover, T_f represents a learning factor that controls the rate of progress of learners.

Each class incorporates a part of the population. In the learning phase, all classes are re-combined into a single population. Inspired by GWO, the top three learners (X_α, X_β, and X_δ) in the population are selected and the rest of the individuals (X_ω) move towards them, D_α, D_β or D_δ denote the distances between the current individual X_i and X_α, X_β, or X_δ, respectively. They are obtained as:

$$D_\alpha = |C_1 \cdot X_\alpha - X_i|, D_\beta = |C_2 \cdot X_\beta - X_i|, D_\delta = |C_3 \cdot X_\delta - X_i| \tag{3}$$

where C_1, C_2, and C_3 are coefficient vectors and they are obtained as:

$$C_1 = 2 \cdot r_1, C_2 = 2 \cdot r_2, C_3 = 2 \cdot r_3 \tag{4}$$

where r_1, r_2, and r_3 are three random values in [0,1].

Therefore, the position of X_i needs to be adjusted towards directions of X_α, X_β, and X_δ. Then, three new vectors, X_1, X_2, X_3 are obtained as:

$$X_1 = X_\alpha - A_1 \cdot D_\alpha, X_2 = X_\beta - A_2 \cdot D_\beta, X_3 = X_\delta - A_3 \cdot D_\delta \tag{5}$$

where A_1, A_2, and A_3 are coefficient vectors obtained as:

$$A_1 = 2a \cdot r_4 - a, A_2 = 2a \cdot r_5 - a, A_3 = 2a \cdot r_6 - a \tag{6}$$

where r_4, r_5, and r_6 are three random values in [0,1], a is a coefficient and linearly decreases from 2 to 0.

Finally, a new position ($X_i(t+1)$) of individual i in iteration $t+1$ is updated as:

$$X_i(t+1) = \frac{X_1 + X_2 + X_3}{3} \tag{7}$$

Moreover, the mutation operation of GA is adopted in MCT. It allows learners to fluctuate in their learning performance, *i.e.*, they may become over-performing and under-performing. Reasonable control of the fluctuation probability (p_m) allows each individual to jump out of local optima. If individual i is fluctuated, its dimension d (X_i^d) is updated as:

$$X_i^d = \check{b}_d + (\hat{b}_d - \check{b}_d) \tag{8}$$

Finally, MCT adopts the Metropolis acceptance rule to select proper individuals for the subsequent optimization process. It allows the selection of individuals whose fitness becomes worse after the update in the next generation [19], which enhances the diversity of the population in new iterations. Therefore, individuals can better cover the decision space, thus improving the exploration ability of MCT. The acceptance probability in our work is calculated as:

$$p = e^{-\frac{\Delta}{T}} \tag{9}$$

where p denotes the acceptance probability and T denotes the starting temperature of SA. Δ denotes the difference in fitness values for each iteration.

B. Autoencoder-assisted Radial Basis Function (ARBF)

Surrogate models aim to substitute the true model to perform function evaluations of individuals. They are applied to reduce the number of expensive fitness evaluations to alleviate the computing overhead of the algorithm. An RBF model can fit functions with a high degree of nonlinearity, and they have great robustness and high convergence speed, which are widely applied as the surrogate model. However, it still suffers from long training time and may become overly complex when facing extensive high-dimensional training data. This not only makes the surrogate model unuseful but also results in the overfitting problem. Therefore, this work considers training the RBF model in a lower-dimensional space. To avoid the problem of dimensional mismatch, the data input for the final prediction also needs to be low-dimensional. Autoencoders are widely adopted as dimension reduction methods, which can model relatively complex nonlinear relationships.

Based on the analysis above, MCT is first adopted to explore the decision space, and positions of the population after each iteration are collected for the training of autoencoder. A trained autoencoder is then used to encode the data for the RBF training. In addition, a linear activation function is adopted for training the RBF model, *i.e.*,

$$\begin{aligned}\gamma(X)&=\varepsilon^{\mathsf{T}}\psi=\sum_{i=1}^{m}\varepsilon_i\psi(X-c_i)\\ \psi_{ij}&=\psi(\|X_i-X_j\|)=\|X_i-X_j\|, i,j=1,\cdots,u\\ \varepsilon&=\psi^{-1}Y\end{aligned} \tag{10}$$

where $\gamma(X)$ denotes the RBF model, $\psi(\cdot)$ denotes the radial basis function, c_i denotes the basis function center i, and m represents the basis function centers' number. Moreover, K-means algorithm [20] is adopted to select the basis function centers. X denotes the input data points, and u denotes their number. $\|\cdot\|$ represents the Euclidian norm. Finally, ε denotes the weight coefficients obtained by X and its set (Y) of corresponding function values.

C. Surrogate-assisted Multi-class Collaborative Teaching and learning optimizer (SMCT)

At the beginning of SMCT, MCT is employed to search in the decision space. Moreover, the positions of each individual in the population are recorded in the database ϕ. It is used to train the autoencoder later. Once the predetermined iteration number is reached, the autoencoder is trained by ϕ. It learns features of the decision space and can be adopted to turn a high-dimensional decision space into a lower-dimensional one. After the autoencoder training, the information of each individual is stored in another database σ for later RBF training. MCT keeps searching until the predetermined iteration number for the RBF training is reached. Afterward, the trained autoencoder is adopted to encode σ into the lower-dimensional data. Then, the K-means algorithm selects basis centers from it, and then the RBF model is constructed. It is worth noting that RBF is trained with low-dimensional data, which significantly reduces its complexity and training time.

Algorithm 1 SMCT

```
Input: $\hat{t}$, maximum number ($\hat{t}_1$) of iterations in MCT for autoencoder training, maximum number ($\hat{t}_2$) of iterations in MCT for ARBF training, database ($\phi$) to train the autoencoder, database ($\sigma$) to train the ARBF, $p_m$, $T$ and reevaluation number ($R$)
Output: Final population ($P$)
1: Initialize population $P$, $\phi=\varnothing$, and $\sigma=\varnothing$
2: while $t\leq\hat{t}$ do
3:     if $t<t_2$ then
4:         $P'=$ MCT($P$)
5:         $f(P')=$FE($P'$)
6:         if $t<t_1$ then
7:             $\phi=\phi\cup P'$
8:         end if
9:         $\sigma=\sigma\cup P'$
10:        Adopt Metropolis acceptance criterion to update $P'$ as $P$
11:        $t=t+1$
12:        if $t=t_1$ then
13:            $A=$ autoencoder($\phi$)
14:            $t=t+1$
15:        end if
16:    else if $t=t_2$ then
17:        $\hat{\sigma}=$ encode($A,\sigma$)
18:        $\gamma=$ARBF($\hat{\sigma}$)
19:        $P'=$ MCT($P$)
20:        $\hat{P'}=$ encode($A,P'$)
21:        $f(P')=$ ARBFpredict($\gamma,\hat{P'}$)
22:        Take the top $C$ learners for reevaluation
23:        Adopt Metropolis acceptance criterion to update $P'$ as $P$
24:        $t=t+1$
25:    else
26:        $P'=$ MCT($P$)
27:        $\hat{P'}=$ encode($A,P'$)
28:        $f(P')=$ ARBFpredict($\gamma,\hat{P'}$)
29:        Take the top $R$ learners for reevaluation
30:        Adopt Metropolis acceptance criterion to update $P'$ as $P$
31:        $t=t+1$
32:    end if
33: end while
34: Return $P$
```

Moreover, during the subsequent optimization process, MCT generates offspring in the decision space. Positions of the population are then encoded by the autoencoder and input to the ARBF, which in turn outputs the predicted fitness values.

Finally, fitness values are predicted by ARBF at the later stage of the optimization process, which may lead to inaccuracy in the final optimization results. As a result, a reevaluation strategy is proposed to solve this problem. Specifically, After each iteration, the top R individuals with better-predicted fitness values are reevaluated by the true model because they have a higher probability of performing better in the true model. Therefore, reevaluating the first few individuals can ensure the accuracy of the search results while saving computational resources. The searching process of SMCT is illustrated in Fig. 1, and the pseudo codes are shown in Algorithm 1. **Autoencoder**($\cdot$) and **RBF**($\cdot$) represent the autoencoder and RBF training. **encode**($\cdot$) represents the encoding phase of the autoencoder. **ARBFpredict** denotes the prediction process of the ARBF. **MCT**($\cdot$) denotes the process of generating offspring by MCT and **FE**($\cdot$) represents the function evaluation of the population. Moreover, the

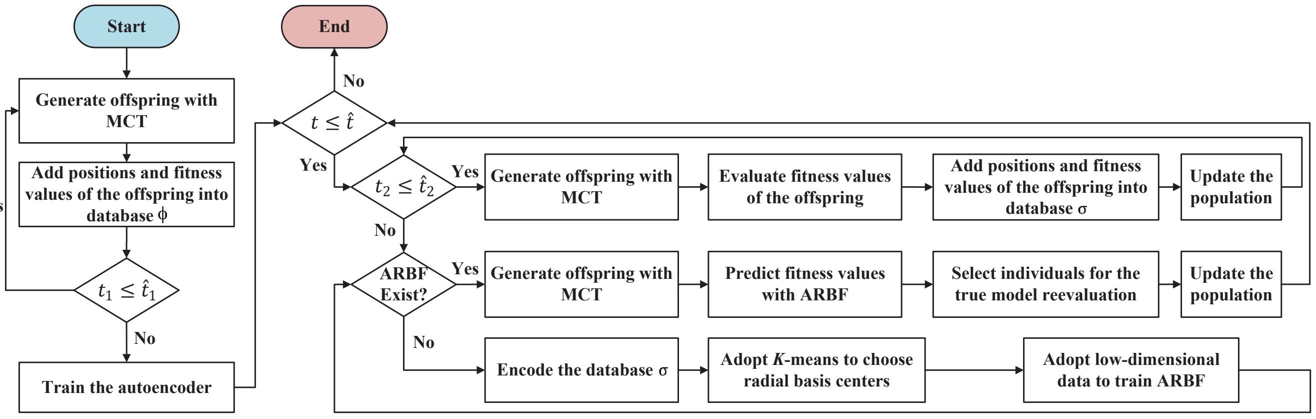
Start
End
Generate offspring with MCT
Add positions and fitness values of the offspring into database ϕ
Yes
$t_1 \leq \hat{t}_1$
No
Train the autoencoder
$t \leq \hat{t}$
No
Yes
$t_2 \leq \hat{t}_2$
Yes
Generate offspring with MCT
Evaluate fitness values of the offspring
Add positions and fitness values of the offspring into database σ
Update the population
No
ARBF Exist?
Yes
Generate offspring with MCT
Predict fitness values with ARBF
Select individuals for the true model reevaluation
Update the population
No
Encode the database σ
Adopt K-means to choose radial basis centers
Adopt low-dimensional data to train ARBF

Fig. 1. Flowchart of SMCT

time complexity of SMCT comprises three parts including autoencoder, RBF, and MCT. The complexity of autoencoder training is $O(qD)$, where q is the sample size for training the autoencoder. The complexity of RBF is $O(uD)$ and that of generating offspring with MCT is $O(\hat{t}DN)$, where $\hat{t}$ is the maximum iteration number of SMCT. Therefore, the complexity of SMCT is $O(qD{+}uD{+}\hat{t}DN)$.

TABLE I
BENCHMARK FUNCTIONS

Functions	D
$F1(x)=\sum_{i=1}^{N} {x_i}^2$	100
$F2(x)=\sum_{i=1}^{N} \lvert x_i\rvert+\prod_{i=1}^{N} \lvert x_i\rvert$	100
$F3(x)=\sum_{i=1}^{N}\left(\sum_{j=1}^{i} x_j\right)^2$	100
$F4(x)=\sum_{i=1}^{N}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	100
$F5(x)=\sum_{i=1}^{N}[x_i^2-10\cos(2\pi x_i)+10]$	100
$F6(x)=-20\exp\left(-0.2\sqrt{\frac{1}{N}\sum_{i=1}^{N} x_i^2}\right)-\exp\left(\frac{1}{N}\sum_{i=1}^{N}\cos(2\pi x_i)\right)+20+e$	100
$F7(x)=\frac{1}{4000}\sum_{i=1}^{N} x_i^2-\prod_{i=1}^{N}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	100
$F8(x)=\frac{\pi}{30}\left\{10sin^2(\pi y_1)+\sum_{i=1}^{N}(y_i-1)^2[1+10sin^2(\pi y_{i+1})]+(y_n-1)^2\right\}$ $+\sum_{i=1}^{N} u(x_i, 10, 100, 4)$	100

III. Comparative Experiments

A. Experimental Settings

Experiments are carried out by comparing SMCT with three typical algorithms including genetic learning particle swarm optimization (GLPSO) [21], TLBO, and Self-adaptive Teaching-learning-based Optimizer with Improved RBF and Sparse Autoencoder (STORA) [22] on eight high-dimensional unimodal and multimodal benchmark functions. The formulas for them are shown in Table I. Moreover, 20 independent executions are carried out for each algorithm for robustness. Moreover, mean values and standard deviations of the final optimization values of each algorithm are recorded. Parameter settings of GLPSO, TLBO, and STORA are given in [11], [21] and [22], respectively. For SMCT, its parameters are set as follows: $\hat{t}$=1000, $\hat{t}_1$=200, $\hat{t}_2$=800, p_m=0.03, T=10^8, T_f=2 and R=5.

TABLE II
EXPERIMENTAL RESULTS

Functions	Algorithms	Mean	Std
F1	**SMCT**	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
	GLPSO	$1.35{\times}10^{+04}$	$2.72{\times}10^{+03}$
	TLBO	$1.35{\times}10^{-163}$	$0.00{\times}10^{+00}$
	STORA	$3.93{\times}10^{-267}$	$0.00{\times}10^{+00}$
F2	**SMCT**	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
	GLPSO	$0.97{\times}10^{+01}$	$0.79{\times}10^{+00}$
	TLBO	$5.17{\times}10^{-85}$	$3.03{\times}10^{-85}$
	STORA	$2.67{\times}10^{-130}$	$3.37{\times}10^{-131}$
F3	**SMCT**	$\mathbf{6.41{\times}10^{-285}}$	$0.00{\times}10^{+00}$
	GLPSO	$4.90{\times}10^{+04}$	$5.05{\times}10^{+03}$
	TLBO	$4.13{\times}10^{+01}$	$5.08{\times}10^{+01}$
	STORA	$9.88{\times}10^{-151}$	$2.20{\times}10^{-150}$
F4	**SMCT**	$\mathbf{9.81{\times}10^{+01}}$	$0.07{\times}10^{+00}$
	GLPSO	$2.47{\times}10^{+03}$	$4.25{\times}10^{+02}$
	TLBO	$9.93{\times}10^{+01}$	$0.49{\times}10^{+00}$
	STORA	$9.88{\times}10^{+01}$	$0.01{\times}10^{+00}$
F5	**SMCT**	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
	GLPSO	$1.25{\times}10^{+02}$	$0.98{\times}10^{+01}$
	TLBO	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
	STORA	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
F6	**SMCT**	$\mathbf{4.44{\times}10^{-16}}$	$3.21{\times}10^{-15}$
	GLPSO	$0.43{\times}10^{+01}$	$0.18{\times}10^{+00}$
	TLBO	$7.99{\times}10^{-15}$	$0.00{\times}10^{+00}$
	STORA	$4.44{\times}10^{-15}$	$3.46{\times}10^{-13}$
F7	**SMCT**	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
	GLPSO	$0.27{\times}10^{+01}$	$0.26{\times}10^{+00}$
	TLBO	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
	STORA	$\mathbf{0.00{\times}10^{+00}}$	$0.00{\times}10^{+00}$
F8	**SMCT**	$\mathbf{1.25{\times}10^{-02}}$	$0.00{\times}10^{+00}$
	GLPSO	$2.13{\times}10^{-02}$	$1.25{\times}10^{-01}$
	TLBO	$2.26{\times}10^{-02}$	$0.01{\times}10^{+00}$
	STORA	$1.31{\times}10^{+00}$	$0.62{\times}10^{+00}$

B. Experimental Results

Fig. 2 shows that SMCT has a faster convergence speed than its peers, and it finds the global optimum at iteration 400. GLPSO has a poor performance on F_1 because it

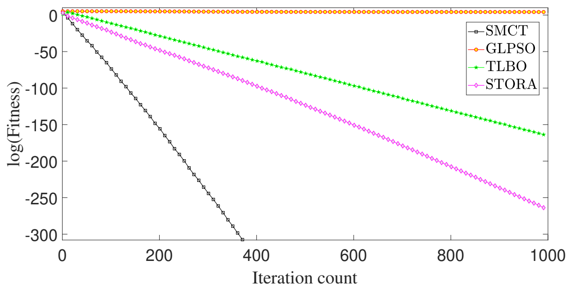

Fig. 2. F1

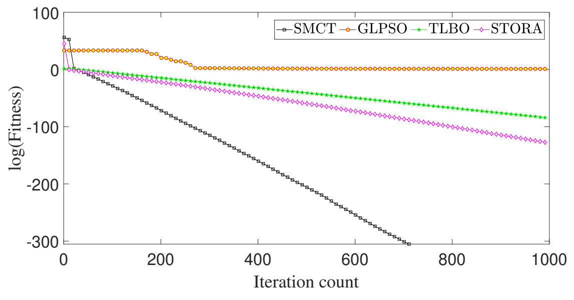

Fig. 3. F2

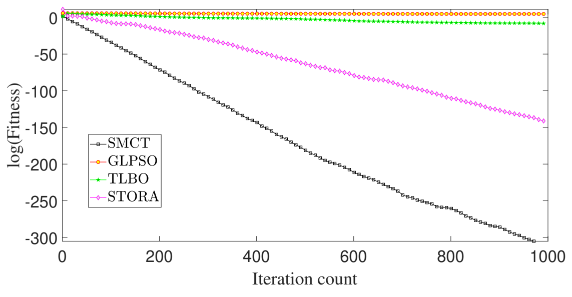

Fig. 4. F3

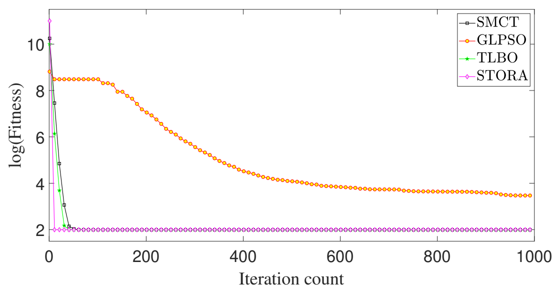

Fig. 5. F4

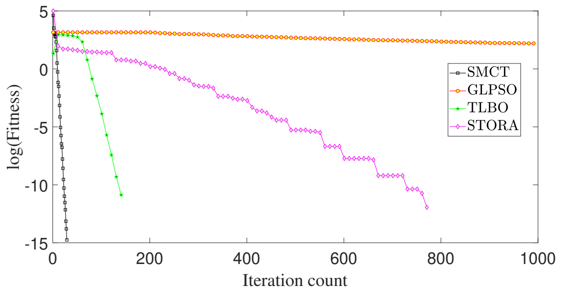

Fig. 6. F5

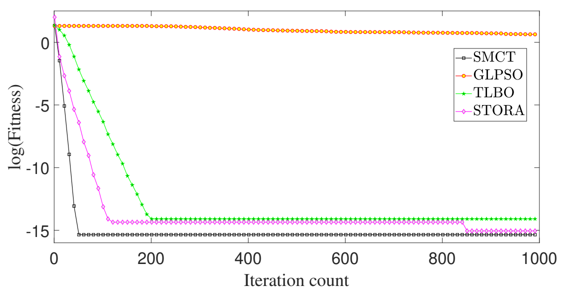

Fig. 7. F6

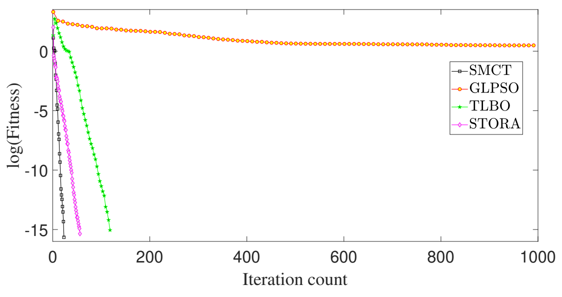

Fig. 8. F7

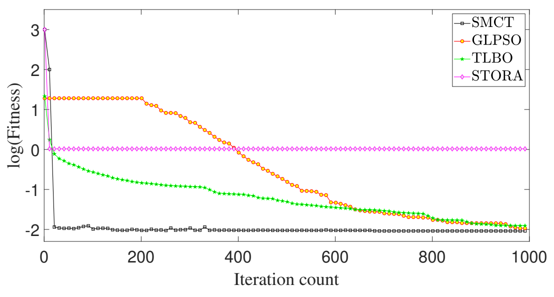

Fig. 9. F8

cannot search the decision space efficiently and falls into local optima. Although TLBO and STORA are not trapped in local optima and both keep moving closer to the global optimum in the decision space, their search speeds are lower than that of SMCT. Figs. 3, 6 and 8 show that F_2, F_5 and F_7 have the similar trend to F_1, SMCT finds the global optimum of benchmark functions and has faster searching speed than its peers. This is because MCT has better search capabilities and search efficiency. It is shown in Fig. 4 that TLBO and GLPSO both trap into a local optima. This is because the landscape features of F_3 are complex. STORA adopts dimensional reduction tools that help it jump out of the local optima and continue to explore the decision space. However, SMCT still has a faster optimization speed and almost finds the global optimum based on Table II. It is illustrated in Fig. 5 that SMCT converges a little later compared with its peers, but it has better optimization results ($9.81 \times 10^{+01}$). Fig. 7 shows that SMCT converges within 100 iterations on F_6. Its convergence speed and final optimization results are both superior to those of its peers. Fig. 9 illustrates that the iteration curve of SMCT on F_8 shows a similar trend to F_6. It is worth noting that the iteration curve shows some fluctuations at the beginning of the search process. This is because SMCT allows fluctuations in learners' performance in the learning phase and adopts SA's Metropolis acceptance criterion. Both of them helps SMCT to jump out of local optima. Finally, it is shown in Table II that SMCT achieves the best fitness values and has small standard deviations on all benchmark functions after all its iterations, which proves its powerful exploration ability and robustness.

IV. CONCLUSIONS

Nowadays, swarm intelligence (SI) algorithms are widely applied in industrial areas, which can optimize industrial processes and bring higher productivity. However, some industrial optimization problems have high-dimensional decision spaces and traditional SI algorithms cannot well solve them. Thus, it is difficult for current algorithms to explore the decision space well, and they may easily fall into local optima. This work proposes an improved Surrogate-assisted Multi-class Collaborative Teaching and learning optimizer (SMCT) for high-dimensional industrial optimization problems. First, to allow its search strategy to better cover the decision space and avoid it from falling into local optima, a multi-class collaborative teaching and learning optimizer is proposed as the base optimizer to enhance search capability in high-dimensional spaces. Then, an autoencoder-assisted radial basis function (ARBF) is designed as a surrogate model to substitute the true model for function evaluations, thereby saving computational resources. Moreover, ARBF adopts an autoencoder to balance its complexity and prediction accuracy. Eight high-dimensional benchmark functions are used to compare our proposed SMCT with three typical algorithms and results show that SMCT has higher search efficiency and better optimization results.

In the future, first, we intend to further apply SMCT to practical high-dimensional optimization problems, *e.g.*, mobile edge computing, and factory scheduling, to test its generalization ability. Second, we will try to employ multiple surrogate models with deep learning simultaneously to collaboratively simulate different regions of the real model, thereby further improving the algorithm.

REFERENCES

[1] H. Lin and C. Tang, "Analysis and Optimization of Urban Public Transport Lines Based on Multiobjective Adaptive Particle Swarm Optimization," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 16786–16798, Sept. 2022.

[2] G. Zhang, B. Liu, L. Wang, D. Yu and K. Xing, "Distributed Co-Evolutionary Memetic Algorithm for Distributed Hybrid Differentiation Flowshop Scheduling Problem," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 5, pp. 1043–1057, Oct. 2022.

[3] Y. Hou, Y. Wu and H. Han, "Multiobjective Differential Evolution Algorithm Balancing Multiple Stakeholders for Low-Carbon Order Scheduling in E-Waste Recycling," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 6, pp. 1912–1925, Dec. 2023.

[4] W. Huang, H. Ding and J. Qiao, "Large-Scale and Knowledge-Based Dynamic Multiobjective Optimization for MSWI Process Using Adaptive Competitive Swarm Optimization," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 54, no. 1, pp. 379–390, Jan. 2024.

[5] J. Bi, Z. Wang, H. Yuan, and J. Zhang, "Cost-Minimized Partial Computation Offloading in Cloud-Assisted Mobile Edge Computing Systems," *2023 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Honolulu, Oahu, HI, USA, 2023, pp. 5052–5057.

[6] P. Wang, K. Li, B. Xiao and K. Li, "Multiobjective Optimization for Joint Task Offloading, Power Assignment, and Resource Allocation in Mobile Edge Computing," *IEEE Internet of Things Journal*, vol. 9, no. 14, pp. 11737–11748, Jul. 2022.

[7] H. Yuan, J. Bi, Z. Wang, J. Yang, and Jia Zhang, "Partial and Cost-minimized Computation Offloading in Hybrid Edge and Cloud Systems," *Expert Systems with Applications*, vol. 250, pp. 1–13, Sept. 2024.

[8] Z. Cheng, J. Ma, X. Li, M. Tomizuka, T. Lee, "Second-Order NonConvex Optimization for Constrained Fixed-Structure Static Output Feedback Controller Synthesis," *IEEE Transactions on Automatic Control*, vol. 67, no. 9, pp. 4854–4861, Sept. 2022.

[9] C. Fu, B. Hou, M. Xue, L. Chang and W. Liu, "Extended Belief Rule-Based System With Accurate Rule Weights and Efficient Rule Activation for Diagnosis of Thyroid Nodules," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 53, no. 1, pp. 251–263, Jan. 2023.

[10] A. Ojha and P. Chanak, "Multiobjective Gray-Wolf-Optimization-Based Data Routing Scheme for Wireless Sensor Networks," *IEEE Internet of Things Journal*, vol. 9, no. 6, pp. 4615–4623, Mar. 2022.

[11] M. Hamzeh, B. Vahidi and A. F. Nematollahi, "Optimizing Configuration of Cyber Network Considering Graph Theory Structure and Teaching–Learning-Based Optimization (GT-TLBO)," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 4, pp. 2083–2090, Apr. 2019.

[12] A. Lensen, B. Xue and M. Zhang, "Genetic Programming for Manifold Learning: Preserving Local Topology," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 4, pp. 661–675, Aug. 2022.

[13] J. Bi, Z. Wang, H. Yuan, J. Zhang, and M. Zhou, "Cost-Minimized Computation Offloading and User Association in Hybrid Cloud and Edge Computing," *IEEE Internet of Things Journal*, doi: 10.1109/JIOT.2024.3354348.

[14] K. Chen, B. Xue, M. Zhang and F. Zhou, "Evolutionary Multitasking for Feature Selection in High-Dimensional Classification via Particle Swarm Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 3, pp. 446–460, Jun. 2022.

[15] M. Cui, L. Li, M. Zhou and A. Abusorrah, "Surrogate-Assisted Autoencoder-Embedded Evolutionary Optimization Algorithm to Solve High-Dimensional Expensive Problems," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 4, pp. 676–689, Aug. 2022.

[16] X. Ji, Y. Zhang, C. He, J. Cheng, D. Gong, X. Gao, Y. Guo, "Surrogate and Autoencoder-Assisted Multitask Particle Swarm Optimization for High-Dimensional Expensive Multimodal Problems," *IEEE Transactions on Evolutionary Computation*, doi: 10.1109/TEVC.2023.3287213.

[17] J. Bi, Z. Wang, H. Yuan, J. Qiao, J. Zhang, and M. Zhou, "Self-adaptive Teaching-learning-based Optimizer with Improved RBF and Sparse Autoencoder for Complex Optimization Problems," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, London, United Kingdom, 2023, pp. 7966–7972.

[18] S. Liu, H. Wang, W. Peng and W. Yao, "A Surrogate-Assisted Evolutionary Feature Selection Algorithm With Parallel Random Grouping for High-Dimensional Classification," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 5, pp. 1087–1101, Oct. 2022.

[19] J. Bi, J. Zhai, H. Yuan, Z. Wang, J. Qiao, J. Zhang, and M. Zhou, "Multi-swarm Genetic Gray Wolf Optimizer with Embedded Autoencoders for High-dimensional Expensive Problems," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, London, United Kingdom, pp. 7265–7271.

[20] S. Guan, Q. Fang and T. Guan, "Application of a Novel PNN Evaluation Algorithm to a Greenhouse Monitoring System," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–12, Jul. 2021.

[21] Y. Gong, J. Li, Y. Zhou, Y. Li, H.Chung, Y. Shi, J. Zhang, "Genetic Learning Particle Swarm Optimization," *IEEE Transactions on Cybernetics*, vol. 46, no. 10, pp. 2277–2290, Oct. 2016.

[22] J. Bi, Z. Wang, H. Yuan, J. Zhang, M. Zhou, "Self-adaptive Teaching-learning-based Optimizer with Improved RBF and Sparse Autoencoder for High-dimensional Problems," *Information Sciences*, vol. 630, pp. 463–481, Jun. 2023.